Triples in the clouds

Zoi Kaoudi, Ioana Manolescu

Triples in the clouds

Zoi Kaoudi and Ioana Manolescu

Inria Saclay–Île-de-France and Université Paris-Sud
Bâtiment 650 (PCRI), 91405 Orsay Cedex, France
firstname.lastname@inria.fr

***Abstract*— The W3C's Resource Description Framework (or RDF, in short) is a promising candidate which may deliver many of the original semi-structured data promises: flexible structure, optional schema, and rich, flexible URIs as a basis for information sharing. Moreover, RDF is uniquely positioned to benefit from the efforts of scientific communities studying databases, knowledge representation, and Web technologies. Many RDF data collections are being published, going from scientific data to general-purpose ontologies to open government data, in particular in the Linked Data movement.**

Managing such large volumes of RDF data is challenging, due to the sheer size, the heterogeneity, and the further complexity brought by RDF reasoning. To tackle the size challenge, distributed storage architectures are required. Cloud computing is an emerging paradigm massively adopted in many applications for the scalability, fault-tolerance and elasticity features it provides. This tutorial discusses the problems involved in efficiently handling massive amounts of RDF data in a cloud environment. We provide the necessary background, analyze and classify existing solutions, and discuss open problems and perspectives.

I. Introduction

During the past decade, many Semantic Web applications have been using the W3C's Resource Description Framework (or RDF, in short) [1] as their data model. RDF data is organized in graphs consisting of *triples* of the form (s, p, o), stating that the subject node s has the property edge p whose value is the object node o. A key concept for RDF is that of *URIs* or Unique Resource Identifiers; these can be used in either of the s, p and o positions to uniquely refer to some entity or concept. Literals (constants) are also allowed in the o position. RDF allows some limited form of incomplete information through *blank nodes*, standing for unknown constants or URIs; an RDF database may, for instance, state that the *author* of X is *Jane* while the *date* of X is *4/1/2011*, for a given, unknown resource X. This contrasts with standard relational databases where all attribute values are either constants or $null$.

RDF Schema (RDFS) [2] is the ontology language of RDF used for giving meaning to resources, grouping them into concepts and identifying the relationships between these concepts. If an RDF Schema is available, RDF semantics requires considering that the database consists not only of triples explicitly present in the store, but also of a set of *implicit triples* obtained through *reasoning based on an RDF Schema and the RDFS rules*. For instance, assume the RDF database contains the fact that the *studentRegistrationNo* of *Bob* is *12345*, whereas an RDF Schema states that only a *student* can have a *studentRegistrationNo*. Then, the fact that *Bob* is a *student* is implicitly present in the database, and a query asking for all *student* instances in the database must return *Bob*.

The proliferation of RDF-based applications has created the need for systems capable of efficiently storing and querying RDF data. Some of the first systems that appeared in the Semantic Web community include Jena [3] and Sesame [4]. More recently, RDF-based stores have gained interest in the database community as well, as illustrated by the works [5], [6], [7]. However, these works focus mostly on RDF viewed as a relational database on which to evaluate conjuctive queries, and do not consider RDF-specific features such as those related to implicit data. Recently, commercial database management systems also started providing some support for RDF, e.g., Oracle 11g RDF [8].

One could consider RDF as yet another graph model for semi-structured data [9], useful in certain application contexts [10] but too expensive to handle in the general case. However, arguably, RDF has the potential for being the most successful semi-structured data model ever, since it draws on a fundamental tenet of the World Wide Web, namely, that every resource is assigned a single URI, which everyone can use to describe it. RDF also draws on other widely adopted W3C specifications, such as XML for serialization and exchange, namespaces for the interoperability of vocabularies etc. Thus, it has good potential for supporting data sharing on the Web.

A particularly interesting class of applications comes from the Open Data concept that "*certain data should be freely available to everyone to use and republish as they wish, without restrictions from copyright, patents or other mechanisms of control*"[1]. Open Data federates players of many roles, from organizations such as business and government aiming to demonstrate transparency and good (corporate) governance, to end users interested in consuming and producing data to share with the others, to aggregators that may build business models around warehousing, curating, and sharing this data [11]. Sample governmental Open Data portals are the ones from the US (`www.data.gov`), UK (`www.data.gov.uk`) and France (`www.etalab.fr`). At the same time, if Open Data designates a general philosophy, Linked Data refers to

[1] http://en.wikipedia.org/wiki/Open_data

the *"recommended best practice for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF"* [12]. In practice, Open and Linked data are frequently combined to facilitate data sharing, interpretation, and exploitation [13]. Sample applications of Linked Open Data are DBPedia (the Linked Data version of Wikipedia), BBC's platform for the World Cup 2010 and the 2012 Olympic games [14] etc.

All these applications have led to plentiful RDF data collections that are available in the Web nowadays. To exploit such large volumes of RDF data, one could try to build a centralized warehouse. However, large and increasing data volumes raise the need for distributed storage architectures. Past works on distributed RDF query processing and reasoning have relied on peer-to-peer platforms [15], [16] or clustered architectures [17], [18], [19].

Cloud computing is an emerging paradigm massively adopted in many applications for the scalability to very large data volumes and the fault-tolerance and elastic allocation that it provides. Recently, interest in massively parallel processing has been renewed by the MapReduce proposal [20] and many follow-up works, which aim at solving large-volume data management tasks based in a cloud environment. For these reasons, cloud-based stores are an interesting avenue to explore for handling very large volumes of RDF data.

This tutorial has two objectives.

- First, we will introduce the audience to the basics of RDF data management, including storage, query processing, reasoning and updating. We will achieve this based on the well-known concepts of semistructured data and existing works on RDF processing of the data management community.
- Second, we will present a classification of the existing architectures and tools for handling large volumes of RDF data in a cloud environment, compare their approaches and algorithms and discuss the respective trade-offs. Finally, we plan to draw a list of problems we currently find open and outline promising avenues to answer them.

II. Tutorial Outline

Our tutorial will be structured as follows.

A. Semistructured data and RDF

We will start by briefly recalling the main principles of *semistructured data* [9], [21], a general concept pioneering many works on complex data management in the database community. The goal is to position RDF as one of the most popular models for semistructured data management currently around, while also acknowledging the contributions previously laid out for the more general model to which it can be traced.

We will provide the necessary *background on RDF processing*. This includes the basic concepts of RDF and RDFS, including their semantics [1]. We will then focus on the Basic Graph Pattern (BGP) queries of SPARQL [22], and in particular on its *conjunctive fragment* allowing to express the core Select-Project-Join database queries. We will introduce the formal semantics of a BGP query, taking into account also the data *entailed* in an RDF database by the presence of an RDFS (or other flavor of) schema.

We will then consider the *core data management issues* raised by RDF processing, namely: storing (in row- and column-oriented stores), indexing, evaluating and ordering joins, query pattern selectivity estimation, updating, and the impact of reasoning. We will devote particular attention to a pedagogic introduction of the issues related to *reasoning*, since they are often ignored in mainstream database works. We will illustrate with detailed examples and introduce the various techniques proposed in the literature to handle implicit data [15], [23] and their performance trade-offs [24], [25].

At the end of this segment, the audience will have a good grasp of the RDF model and RDF data management issues.

We will end this part of the tutorial by noting that duplicate elimination is an important issue for RDF, since the usage of distinct URIs for the same conceptual resource hinders the basic RDF triple linking mechanism (through shared URIs). A full discussion of this issue is out of our scope, but we will mention it and provide pointers to the relevant literature [26], [27].

B. Cloud-based data management

Interest in massively parallel processing has been renewed recently since the emergence of the MapReduce proposal [20] and its open source implementation Hadoop [28]. MapReduce has become popular in various computer science fields as it provides a simple programming paradigm which frees the developer from the burden of handling the issues of parallelization, scalability, load balancing and fault-tolerance. Although MapReduce was first mostly intended for data analysis tasks, it has also started to be used in query processing tasks. However, MapReduce provides simple primitives and more complex operations such as joins are not directly supported.

In this part of the tutorial, we will recall the basics of MapReduce and briefly outline the main existing strategies for processing joins in a MapReduce framework [29], [30], [31], since joins are at the heart of RDF BGP query processing.

RDFS reasoning can be assimilated to deductive databases and therefore recursive query processing techniques are pertinent. Therefore, we will highlight the connections between cloud-based RDFS reasoning and recursive query processing on top of MapReduce [32], [33].

This part of the tutorial will have introduced the audience to the basic primitives available in the cloud, the strong advantages of MapReduce and the difficulties that remain to be solved for high-level, declarative management of complex data such as RDF.

C. State-of-the-art cloud-based RDF systems

The core part of our tutorial will be a comprehensive classification of existing architectures and systems for handling RDF data within a cloud. We will present the main priciples used for: (i) organizing the data store, (ii) processing conjunctive queries and (iii) handling implicit data through reasoning.

A first classification of existing platforms can be made according to their underlying *data storage* facilities:

- The first category includes systems which use existing "NoSQL" key-value stores [34] as their back-end for storing and indexing RDF data;
- Our second category consists of systems relying on a distributed file system, such as HDFS, for warehousing RDF data;
- The third category comprises systems relying on other storage facilities, such as a set of independent single-site RDF stores, or data storage services supplied by the cloud providers.

Representatives of the first category include systems such as Rya [35] which uses Apache Accumulo [36], Cumulus-RDF [37] based on Apache Cassandra [38], Stratustore [39] which relies on Amazon's SimpleDB [40], and H_2RDF [41] which is built on top of HBase [42]. The second category comprises platforms such as those described in [43], [44]. These systems are built to make the most out of the parallel processing capacities provided by the underlying MapReduce paradigm. However they may be seen disadvantaged from the perspective of the data store, given that they do not have efficient fine-grained data stores to rely on. Within the third category, centralized RDF stores distributed among multiple nodes are used in [45], whereas [46], [47] use a mixed approach with data residing in Amazon's storage service (S3) and a full data index built in Amazon's key-value store.

Cloud-based RDF platforms can also be analyzed according to their strategy for *processing conjunctive queries*:

- systems relying on the parallel programming paradigm of MapReduce [20];
- systems attempting to reduce or avoid alltogether MapReduce steps. The reason is that while MapReduce achieves important parallel scalability, the start-up time of a MapReduce job is significant and it may be too high for interactive-style queries.

Systems such as [43], [44] belong to the first class above. In the second class we find systems relying on key-value stores, typically implementing their own join operators, often exploiting the key-based organization of the underlying store, which helps identical values from different triples meet in the same data partitions [35], [39]. Works such as [45], [46], [47] which take advantage of existing RDF stores also belong to this group. Finally, [41] employs a hybrid approach depending on the nature of the query.

Having toured the issues of RDF storing and querying through the above techniques, we will show how *reasoning* is handled in these cloud-based systems. From this angle, the options are as follows:

- pre-compute and materialize all implicit triples;
- compute the necessary implicit triples at query time;
- some hybrid approach among the two above, with some implicit data computed statically and some at query time.

Most recent proposals on RDFS reasoning in cloud environments come from the Semantic Web community [48], [49] but do not integrate it with the querying phase. The only work from the above that injects RDF reasoning within query processing is [43].

This part of the tutorial will provide a principled categorization of existing works and point out the strength and limitations of various proposals.

D. Open issues

In the final part of the tutorial we will draw a list of problems we currently find open and outline some promising avenues to answer them. These range from RDF query optimization and RDF updates in a cloud environment, as well as RDF view materialization and sharing of intermediate results in the distributed environment of the cloud.

Targeted Audience The intended audience consists of database students and researchers with an interest in RDF data management, cloud-based data processing, or both. We will provide brief self-sufficient introductions to both these areas to enable non-specialists to follow. In particular, while RDF querying has been well explored in database-oriented works, we will devote some time to the issues involved in RDF reasoning, which have not been considered in these works.

Acknowledgments We have studied the material on which the tutorial is based, while collaborating with many colleagues. We wish to thank in particular François Goasdoué, Alexandra Roatiş and Jorge Quiané-Ruiz for many insightful discussions.

III. Presenters

Zoi Kaoudi received her PhD from National and Kapodistrian University of Athens in 2011, after graduating from the school of Electrical and Computer Engineering of the National Technical University of Athens. She is now a postdoctoral researcher at Inria Saclay and member of the OAK team. Her research interests include cloud-based management of Web data, distributed RDF query processing and reasoning. Personal webpage: `http://pages.saclay.inria.fr/zoi.kaoudi/`

Ioana Manolescu received her PhD from Inria and Université de Versailles Saint-Quentin in 2001, after graduating from Ecole Normale Supérieure in Paris. Ioana has been a post-doc at Politecnico di Milano, Italy, then she joined Inria where she is now senior researcher and the leader of the OAK team, specialized in database optimization techniques for complex, large data. Her research interests include algebraic optimizations, adaptive storage and efficient management of semantically rich data and cloud-based management of Web data. She has previously presented tutorials in VLDB (2003), ICDE (2005 and 2008), and the EDBT summer school (2002 and 2004). Personal webpage: `http://pages.saclay.inria.fr/ioana.manolescu/`

References

[1] P. Hayes, "RDF Semantics," W3C Recommendation, February 2004, http://www.w3.org/TR/rdf-mt/.

[2] D. Brickley and R. Guha, "RDF Vocabulary Description Language 1.0: RDF Schema," W3C Recommendation, Tech. Rep., 2004.

[3] K. Wilkinson, C. Sayers, H. A. Kuno, and D. Raynolds, "Efficient RDF Storage and Retrieval in Jena2," in *1st International Workshop on Semantic Web and Databases (SWDB), in conjunction with VLDB*, 2003.

[4] J. Broekstra and A. Kampman, "Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema," in *1st International Semantic Web Conference (ISWC)*, ser. LNCS, vol. 2342, 2002.

[5] T. Neumann and G. Weikum, "The RDF-3X Engine for Scalable Management of RDF Data," *VLDBJ*, vol. 19, no. 1, 2010.

[6] D. J. Abadi, A. Marcus, S. Madden, and K. Hollenbach, "SW-Store: a vertically partitioned DBMS for Semantic Web data management," *VLDB J.*, vol. 18, no. 2, pp. 385–406, 2009.

[7] C. Weiss, P. Karras, and A. Bernstein, "Hexastore: sextuple indexing for semantic web data management," *PVLDB*, vol. 1, no. 1, pp. 1008–1019, 2008.

[8] E. I. Chong, S. Das, G. Eadon, and J. Srinivasan, "An efficient SQL-based RDF querying scheme," in *VLDB*, 2005.

[9] D. Quass, J. Widom, R. Goldman, K. Haas, Q. Luo, J. McHugh, S. Nestorov, A. Rajaraman, H. Rivero, S. Abiteboul, J. Ullman, and J. Wiener, "Lore: a lightweight object repository for semistructured data," in *SIGMOD*, 1996.

[10] S. Trißl and U. Leser, "Fast and practical indexing and querying of very large graphs," in *SIGMOD*, 2007.

[11] G. Raschia, M. Theobald, and I. Manolescu, "Proceedings of the first international workshop on open data (WOD)," 2012.

[12] T. Berners-Lee, "Linked data - design issues," http://www.w3.org/DesignIssues/LinkedData.html, 2006.

[13] "State of the LOD cloud," Available from: http://www4.wiwiss.fu-berlin.de/lodcloud/state/, 2011.

[14] A. Kiryakov, B. Bishoa, D. Ognyanoff, I. Peikov, Z. Tashev, and R. Velkov, "The Features of BigOWLIM that Enabled the BBC's World Cup Website," in *Workshop on Semantic Data Management*, 2010.

[15] Z. Kaoudi, I. Miliaraki, and M. Koubarakis, "RDFS Reasoning and Query Answering on Top of DHTs," in *7th International Conference on The Semantic Web (ISWC)*, 2008.

[16] Z. Kaoudi, K. Kyzirakos, and M. Koubarakis, "SPARQL Query Optimization on Top of DHTs," in *9th International Conference on The Semantic Web (ISWC)*, 2010.

[17] S. Harris, N. Lamb, and N. Shadbolt, "4store: The Design and Implementation of a Clustered RDF Store," in *5th International Workshop on Scalable Semantic Web Knowledge Base Systems*, 2009.

[18] A. Owens, A. Seaborne, N. Gibbins, and M. Schraefel, "Clustered TDB: A Clustered Triple Store for Jena," Technical Report, 2008.

[19] O. Erling and I. Mikhailov, "RDF Support in the Virtuoso DBMS," *Networked Knowledge - Networked Media*, pp. 7–24, 2009.

[20] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," in *OSDI*, 2004.

[21] Y. Papakonstantinou, H. Garcia-Molina, and J. Widom, "Object exchange across heterogeneous information sources," in *ICDE*, 1995.

[22] E. Prud'hommeaux and A. Seaborn, "SPARQL Query Language for RDF," W3C Recommendation, http://www.w3.org/TR/rdf-sparql-query/, 2008.

[23] S. Abiteboul, I. Manolescu, P. Rigaux, M.-C. Rousset, and P. Senellart, *Web Data Management and Distribution*. Cambridge University Press, Dec 2011.

[24] F. Goasdoué, I. Manolescu, and A. Roatiş, "BGP Query Answering against Dynamic RDF Databases," Inria Research Report no. 8018, 2012.

[25] ——, "Getting more RDF support from relational databases (poster)," in *World Wide Web (WWW) Conference*, 2012.

[26] F. Naumann and M. Herschel, *An Introduction to Duplicate Detection*, ser. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2010.

[27] C. Bohm, G. de Melo, F. Naumann, and G. Weikum, "LINDA: Distributed web-of-data-scale entity matching (short paper)," in *21st International Conference on Information and Knowledge Management (CIKM)*, 2012.

[28] "Apache Hadoop," http://hadoop.apache.org/.

[29] S. Blanas, J. M. Patel, V. Ercegovac, J. Rao, E. J. Shekita, and Y. Tian, "A Comparison of Join Algorithms for Log Processing in MapReduce," in *SIGMOD*, 2010.

[30] F. N. Afrati and J. D. Ullman, "Optimizing Multiway Joins in a Map-Reduce Environment," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 9, 2011.

[31] ——, "Optimizing Joins in a Map-Reduce Environment," in *EDBT*, 2010.

[32] F. N. Afrati, V. R. Borkar, M. J. Carey, N. Polyzotis, and J. D. Ullman, "Map-Reduce Extensions and Recursive Queries," in *EDBT*, 2011.

[33] Y. Bu, B. Howe, M. Balazinska, and M. D. Ernst, "The HaLoop Approach to Large-Scale Iterative Data Analysis," *VLDB J.*, vol. 21, no. 2, 2012.

[34] R. Cattell, "Scalable SQL and NoSQL data stores," *SIGMOD Record*, vol. 39, no. 4, pp. 12–27, May 2011.

[35] R. Punnoose, A. Crainiceanu, and D. Rapp, "Rya: A Scalable RDF Triple Store for the Clouds," in *1st International Workshop on Cloud Intelligence (in conjunction with VLDB)*, 2012.

[36] "Apache Accumulo," http://accumulo.apache.org/.

[37] G. Ladwig and A. Harth, "CumulusRDF: Linked Data Management on Nested Key-Value Stores," in *SSWS*, 2011.

[38] "Apache Cassandra," http://cassandra.apache.org/.

[39] R. Stein and V. Zacharias, "RDF On Cloud Number Nine," in *4th Workshop on New Forms of Reasoning for the Semantic Web: Scalable and Dynamic*, May 2010.

[40] "Amazon Web Services," http://aws.amazon.com/.

[41] N. Papailiou, I. Konstantinou, D. Tsoumakos, and N. Koziris, "H2RDF: adaptive query processing on RDF data in the cloud," in *21st international conference on World Wide Web (demo)*, 2012.

[42] "Apache HBase," http://hbase.apache.org/.

[43] M. Husain, J. McGlothlin, M. M. Masud, L. Khan, and B. M. Thuraisingham, "Heuristics-Based Query Processing for Large RDF Graphs Using Cloud Computing," *IEEE Trans. on Knowl. and Data Eng.*, 2011.

[44] K. Rohloff and R. E. Schantz, "High-Performance, Massively Scalable Distributed Systems using the MapReduce Software Framework: the SHARD Triple-Store," in *Programming Support Innovations for Emerging Distributed Applications*, 2010.

[45] J. Huang, D. J. Abadi, and K. Ren, "Scalable SPARQL Querying of Large RDF Graphs," *PVLDB*, vol. 4, no. 11, pp. 1123–1134, 2011.

[46] F. Bugiotti, F. Goasdoué, Z. Kaoudi, and I. Manolescu, "RDF Data Management in the Amazon Cloud," in *Data Analytics in the Cloud (DanaC) Workshop (in conjunction with EDBT)*, 2012.

[47] A. Aranda-Andújar, F. Bugiotti, J. Camacho-Rodríguez, D. Colazzo, F. Goasdoué, Z. Kaoudi, and I. Manolescu, "Amada: Web Data Repositories in the Amazon Cloud (demo)," in *21st International Conference on Information and Knowledge Management (CIKM)*, 2012.

[48] J. Urbani, S. Kotoulas, E. Oren, and F. van Harmelen, "Scalable Distributed Reasoning using MapReduce," in *8th International Semantic Web Conference (ISWC)*, October 2009.

[49] J. Urbani, F. van Harmelen, S. Schlobach, and H. Bal, "QueryPIE: Backward Reasoning for OWL Horst over Very Large Knowledge Bases," in *10th International Semantic Web Conference (ISWC)*, 2011.